

## 前提

- ▶ 定型手順はこの流れで細かいところを調整ぐらいで使えるように
- ▶ 開発マシンと本番マシンは別の想定。
- ▶ 開発マシンではdevelopment環境、本番マシンではproduction環境を使用。
- ▶ 開発マシンから本番マシンへのデプロイはcapistranoを使う。
- ▶ ソースコード管理はsubversionを使う。リポジトリはsubversionサーバ用のマシンに置く想定。
- ▶ 開発マシンではscript/serverで動かす。本番マシンではフロントをApache+mod\_ploxy\_balancerで、バックをmongrel\_cluster構成とする。
- ▶ 開発マシンにはNetBeans6.0以上にrubyプラグイン入れとけ。個人的にはAptenIDE+Radrailsより、NetBeansの方が上かと。

## リポジトリを用意する

subversionのサーバにログインして

```
$ svnadmin create --fs-type fsfs /svn/hogeproject
```

んで、/svn/hogeproject/confの中のsvnserve.confとかpasswdあたりをいじっておく。

## railsアプリを作ってsubversionに入れておく

まずはsubversion用のディレクトリにアプリを生成し、コミットする。

```
$ mkdir hogeproject
$ cd hogeproject
$ mkdir trunk tags branches
$ cd trunk
$ rails hogeapp
$ cd ..
$ svn import . svn://domain/svn/hogeproject/ -m "initial" --username user
```

登録できたら、元ディレクトリをどけて(バックアップとして)、チェックアウトする。

```
$ mv mv hogeproject hogeproject_backup
$ svn checkout svn://domain/svn/hogeproject/trunk/ .
```

logとtmpのいらぬファイルを無視する

```
$ cd hogeapp
$ svn remove ./log/*
$ svn commit -m "remove log file"
$ svn propset svn:ignore "*.log" log/
$ svn update log/
$ svn commit -m "add ignore log/*"
$ svn remove tmp/*
$ svn propset svn:ignore "*" tmp/sessions tmp/cache tmp/sockets
$ svn update tmp/
```

```
$ svn commit -m "add ignore tmp/"
```

## 先に入れておくべきgem

```
$ gem install mysql gettext refe mongrel mongrel_cluster capistrano rak rapt rspec rocketstarter sqld4r  
sqlite3-ruby -y
```

gemのgettextはGNU gettextのバインディングではなく、再実装らしいので、gettextのライブラリ(yum install gettext  
みたいな)は必要ないみたい。

## なにはなくとも

./config/environment.rbに

```
$KCODE = 'u'  
require 'jcode'  
require 'gettext/rails'
```

を追加

app/controllers/application.rbのクラス内に

```
init_gettext "app-name" # ここはrakeファイル内のMY_APP_TEXT_DOMAINと一致させる必要がある
```

を追加

```
require 'rubygems'  
require 'gettext/utils'  
desc 'Update pot/po files.'  
task :updatepo do
```

```
MY_APP_TEXT_DOMAIN = "app-name" # ここはinit_gettextで指定したテキストドメインと一致させる必要がある  
MY_APP_VERSION     = "app-name 1.0.0"  
GetText.update_pofiles(MY_APP_TEXT_DOMAIN,  
                        Dir.glob("{app,lib}/**/*.{rb,rhtml}"),  
                        MY_APP_VERSION)end
```

```
end  
desc 'Create mo-files'  
task :makemo do
```

```
GetText.create_mofiles(true, 'po', 'locale')
```

```
end
```

po/mo用のrakeタスクをRakefileに追加

rake makepoしたら

```
mkdir po/ja
cp po/hoge.pot po/ja/hoge.po
```

として編集。ディレクトリ名は日本語ならja, jpとしたら泣けるはず。

翻訳に飽きたら

```
rake makemo
```

しておく。

./config/database.ymlのdevelopmentとtestとproductionに

```
encoding: UTF8
```

と各種設定情報を追加

```
$ capify
```

して./config/deploy.rbを埋めておく。

[Capistrano](#)参照。

```
$ mongrel_rails cluster::configure
```

して./config/mongrel\_cluster.rbに本番環境用の設定を埋めておく。(cap deployでspinを使うなら必要無い)

memcached使いたかったら

```
sudo gem install memcache-client
```

で入れて、config/production.rbに

```
config.action_controller.fragment_cache_store = :mem_cache_store
config.action_controller.session_store = :mem_cache_store
```

を追加する。development.rbには入れなくてよいかもしれない。

ちなみにフロントエンドのapache 2.2系でmod\_poxy\_balancerを使った構成は

```
NameVirtualHost 192.168.1.3:80

<VirtualHost 192.168.1.3:80>
  DocumentRoot "/sites/yourdomain.com/current/public"
  ServerName yourdomain.com
  CustomLog /var/log/apache2/yourdomain.com.proxy_access.log combined
  ErrorLog /var/log/apache2/yourdomain.com.error.log
  <Directory "/sites/yourdomain.com/current/public">
    allow from all
  Options +Indexes FollowSymLinks
  AllowOverride all
  Allow from all
  Order allow,deny
</Directory>
```

```
DirectoryIndex index.html

# forward proxy off
ProxyRequests Off

# don't use reverse-proxy for /engine_files /images /javascripts /stylesheets
ProxyPass /engine_files !
ProxyPass /images !
ProxyPass /javascripts !
ProxyPass /stylesheets !

# other access are proxying mongrel server
ProxyPass / balancer://yourcluster/ timeout=2 nofailover=On
ProxyPassReverse / http://localhost:4000/
ProxyPassReverse / http://localhost:4001/
</VirtualHost>

#reverse
proxy cluster member
<Proxy balancer://yourcluster>
BalancerMember http://localhost:4000 loadfactor=10
BalancerMember http://localhost:4001 loadfactor=10
</Proxy>

<Location /balancer-manager>
SetHandler balancer-manager
Order Deny,Allow
Deny from all
Allow from 192.168.1.2
</Location>

<Location /server-status>
SetHandler server-status
Order Deny,Allow
Deny from all
Allow from 192.168.1.2
</Location>

ExtendedStatus On
```

こんな感じ。

192.168.1.3が本番マシンのIPで、

192.168.1.2は本番マシンをリモートメンテするためのマシンのIPを想定。

詳しくはapacheのマニュアル読むべし。

ただし現在は[Passenger](#)がオススメ

## その他雑記・メモ

- ▶ AR.update\_attributesでの一括アップデート時に、特定カラムを保護する

```
attr_protected :brockcolumn
```

- ▶ HTTP headerを書き換える

```
@response.headers['name'] = value
@headers['name'] = value
```

- ▶ キャラクタセット変換(HTML headerは自分でなんとかすること)

```
before_filter :set_characterstet
after_filter :translate_body
```

```
def set_characterstet
  @headers['Content-Type'] = 'text/html; charset=Shift-JIS'
end
def translate_body
  @response.body = NKF.nkf('-Ws -m0 -x params for nkf', @response.body)
end
```

- ▶ 共用railsサーバ用のログローテーション設定(細かい設定理解してないので注意)

/etc/logrotate.d/rails\_domains

```
/var/www/domains/*/*/shared/log/* {
```

```
  weekly
  rotate 4
  copytruncate
  compress
  notifempty
  missingok
```

```
}
```

- ▶ crlfをbrタグに置き換える

application\_helperなどで

```
def hbr(str)
  str = html_escape(str)
  str.gsub(/\r\n|\r|\n/, "<br />")
end
```

として、テンプレートで

```
<%= hbr @log.text %>
```

- ▶ 行毎にCSSなどを切り替える

cycleメソッドを使えばよい。引数の数を変えれば3回に1回なども可能

```
<table>
<% for user in @users %>
```

```
<tr class="<%= cycle "even", "odd" %>">  
  <td>~ ~ ~</td>  
</tr>
```

```
<% end %>  
</table>
```